OIPE JCS
AUG 2 3 2004
PATENT & TRADEMARK OFFICE

AF
IFW

Our Docket No: 2950P055 Patent

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

In re the Patent Application of:)
)
Barnett et al.)
)
Serial No.: 09/583,311) Art Unit: 2124
)
Filed: May 30, 2000)
) Examiner: Roche, Trenton J.
For: Method and Apparatus for)
Automating Testing of Java Beans)

Mail Stop Appeal Brief – Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

APPEAL BRIEF IN SUPPORT OF APPELLANTS' APPEAL TO THE BOARD OF PATENT APPEALS AND INTERFERENCES

Sir:

Applicants (hereinafter "Appellants") hereby submit this Brief in triplicate in support of its appeal from a final decision by the Examiner, mailed March 23, 2004, in the above-captioned case. Appellants respectfully request consideration of this appeal by the Board of Patent Appeals and Interferences (hereinafter "Board") for allowance of the above-captioned patent application.

An oral hearing is not desired.

FIRST CLASS CERTIFICATE OF MAILING

I hereby certify that I am causing the above-referenced correspondence to be deposited with the United States Postal Service as first class mail with sufficient postage on the date indicated below and that this paper or fee has been addressed to Mail Stop Appeal Brief – Patents, Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.

Date of Deposit: August 19, 2004
Name of Person Mailing Correspondence: Debbie Peloquin

Debbie Peloquin
Signature

August 19, 2004
Date

08/23/2004 SSESHE1 00000098 09583311
01 FC:1402 330.00 OP

TABLE OF CONTENTS

I. REAL PARTY IN INTEREST 3

II. RELATED APPEALS AND INTERFERENCES 3

III. STATUS OF THE CLAIMS 3

IV. STATUS OF AMENDMENTS 3

V. SUMMARY OF THE INVENTION 3

VI. ISSUE PRESENTED 5

VII. GROUPING OF CLAIMS 5

VIII. ARGUMENT 5

IX. CONCLUSION 10

X. APPENDIX OF CLAIMS i

I. REAL PARTY IN INTEREST

The invention is assigned to Aspect Communications Corporation of 1310 Ridder Park Drive, San Jose, California 95131-2313.

II. RELATED APPEALS AND INTERFERENCES

To the best of Appellants' knowledge, there are no appeals or interferences related to the present appeal that will directly affect, be directly affected by, or have a bearing on the Board's decision.

III. STATUS OF THE CLAIMS

Claims 1, 4-13, 16-25, and 28-36 are currently pending in the above-referenced application. No claims have been allowed. Claims 1, 4-13, 16-25, and 28-36 stand rejected as being unpatentable under U.S. Patent No. 6,473,894 of Shrader, et al. ("Shrader") in view of U.S. Patent No. 6,457,142 of Klemm, et al. ("Klemm") in the final Office Action, mailed March 23, 2004, and are the subject of this appeal.

IV. STATUS OF AMENDMENTS

In response to the final Office Action, mailed on March 23, 2004, rejecting claims 1, 4-13, 16-25, and 28-36, Appellants timely filed a Notice of Appeal on June 22, 2004.

A copy of all claims on appeal is attached hereto as Appendix of Claims.

V. SUMMARY OF THE INVENTION

Object oriented programming languages such as Java are frequently used in a variety of applications. These applications include programs for use on personal computers as well as complex systems such as those used in Computer Telephony Integration (CTI) networks. In a CTI system, applications written using Java interface with a CTI server to interact with a switch or Automatic Call Distributor (ACD). Object oriented languages such as Java are popular for use in such applications because of their ease of maintenance and expandability. However, testing of new or revised software is a time consuming and therefore expensive process.

In order to test a new or revised program, a programmer is generally required to write a test script or test program which is specific to a particular switch. Such a script or program may include commands to test the functionality of a switch, and possibly provide means for outputting results of the test. The technique of writing a switch specific testing program is time consuming and therefore costly.

In one embodiment, a method for testing a first program includes reading an input file containing a location of a first program, identifiers of other programs to be invoked by the first program, and arguments to be passed to other programs. The input file further includes special commands having a loop command and a sleep command. The loop command is to cause execution of the first program including repeating of instructions for a number of iterations until occurrence of an endloop instruction. The sleep command is to cause execution of the first program to pause for a specified time. Further, the execution of the first program includes invoking the other programs and passing the arguments to the other program, and generating one or more log files based on results of the execution of the first program. (see claim 1; see also Background of the Invention, page 2, Summary of the Invention, page 3).

VI. ISSUE PRESENTED

Whether the present invention as recited by claims 1, 4-13, 16-25, and 28-36 is unpatentable under Shrader in view of Klemm.

VII. GROUPING OF CLAIMS

For the purposes of this appeal, the claims are grouped as follows:

Claims 1, 4-13, 16-25, and 28-36 stand and fall together.

VIII. ARGUMENTS

Claims 1, 4, 6, 8-10, 13, 16, 18, 20-22, 25, 28, 30, and 32-34 stand rejected under 35 U.S.C. § 103 as being as being unpatentable under Shrader in view of Klemm as stated in the final Office Action, mailed March 23, 2004.

Claims 7, 11, 19, 23, 31 and 35 stand rejected under 35 U.S.C. §103(a), as being unpatentable over Shrader in view of Klemm further in view of Barskiy et al., U.S. Patent No. 6,205,412 ("Barskiy") as stated in the final Office Action, mailed March 23, 2004.

Claims 5, 12, 17, 24, 29 and 36 stand rejected under 35 U.S.C. §103(a), as being unpatentable over Shrader in view of Klemm further in view of Logan et al., U.S. Patent No. 6,601,018 ("Logan") as stated in the final Office Action, mailed March 23, 2004.

A. BECAUSE SHRADER AND KLEMM, NEITHER INDIVIDUALLY NOR WHEN COMBINED, TEACH OR REASONABLY SUGGEST AN INPUT FILE HAVING A LOOP COMMAND TO CAUSE EXECUTION OF THE FIRST PROGRAM INCLUDING REPEATING OF INSTRUCTIONS FOR A NUMBER OF ITERATIONS

UNITL OCCURRENCE OF AN ENDLOOP INSTRUCTION, THE REJECTION OF THE CLAIMS UNDER 35 U.S.C. § 103 IS IMPROPER.

Claim 1, recites:

> A method of testing a first program, comprising:
> reading an input file containing a location of a first program, identifiers of other programs to be invoked by the first program, and arguments to be passed to the other programs, wherein the input file further includes special commands having
> a loop command to cause execution of the first program including repeating of instructions for a number of iterations until occurrence of an endloop instruction, and
> a sleep command to cause execution of the first program to pause for a specified time;
> executing the first program including invoking the other programs and passing the arguments to the other programs; and
> generating one or more log files based on results of the execution of the first program.
> (emphasis provided)

Shrader discloses "read[ing] test input file . . . [to] create URL test array" for "testing of applets or application in a data processing system" (fig. 3; col. 1, lines 8-10; col. 4, lines 39-41). Shrader further discloses "URL test array is created by test/run program *when it reads test input file . . . contain[ing] the URLs and parameters* for testing, but *may not contain a number of times for repeating the test or execution*, which *may instead default to a predetermined number*" (col. 4, lines 39-52; emphasis provided). First, at best, Shrader discloses having an *input file to default to a predetermined number* which is not the same as using a loop command for repeating of instructions until an endloop instruction has occurred, as recited by claim 1.

Second, Shrader's "loop counter variable . . . [is] employed to *store the number of times each source page has been reloaded* . . . [because] each source page is to be loaded in sequence before any source page is reloaded" *after the creation of the URL test array* using the "elements and entities within the test array" (fig. 3; col. 7, lines 12-15; col. 7, lines 25-28; emphasis provided). Stated differently, Shrader's loop counter variable uses the elements and entities of the already created test array to store the number of times each source page has been loaded. Shrader does not teach or reasonably suggest the input file having a loop command to cause execution of the first program including repeating of instructions for a number of iterations until occurrence of an endloop instruction, as recited by claim 1.

Klemm discloses a "fault monitoring, performance monitoring and fault tolerance apparatus and method for target application programs is realized in an application supervisor by employing a supervisor agent, modified application programming interfaces (APIs), a generic application wrapper and *a shell script that operate interactively to detect and automatically resolve reliability and performance problems* occurring in executing the target application program" (Abstract; see col. 1, lines 29-38; emphasis provided). Klemm discloses a *shell script* and, like Shrader, does not teach or reasonably suggest the input file having a loop command to cause execution of the first program including repeating of instructions for a number of iterations until occurrence of an endloop instruction, as recited by claim 1.

Hence, Shrader and Klemm, neither individually nor when combined, teach or reasonably suggest the input file having a loop command to cause execution of the first program including repeating of instructions for a number of iterations until occurrence of an endloop instruction, as recited by claim 1. Accordingly, for at least the reasons set forth above, Appellants respectfully submit that claim 1 and its dependent claims are allowable over the cited references.

With regard to independent claims 13 and 25, these claims contain limitations similar to those of claim 1. Accordingly, Appellants respectfully submit that claims 13 and 25 and their dependent claims are allowable over the cited references.

With regard to claims 5, 7, 11-12, 17, 19, 23-24, 29, 31 and 35-36, these claims depend from one of independent claims 1, 13 and 25 and thus include the limitations of the independent claim from which they depend. Accordingly, Appellants respectfully submit that claims 5, 7, 11-12, 17, 19, 23-24, 29, 31 and 35-36 are allowable over the cited references.

B. BECAUSE SHRADER AND KLEMM, NEITHER INDIVIDUALLY NOR WHEN COMBINED, TEACH OR REASONABLY SUGGEST AN INPUT FILE HAVING A SLEEP COMMAND TO CAUSE EXECUTION OF THE FIRST PROGRAM TO PAUSE FOR A SPECIFIED TIME, THE REJECTION OF THE CLAIMS UNDER 35 U.S.C. § 103 IS IMPROPER.

As acknowledged by the Examiner, Shrader does not teach or reasonably suggest a sleep command to cause execution of the first program to pause for a specified time as recited by claim 1 (Office Action, mailed March 23, 2004, page 4). Klemm discloses a "*periodic thread check* . . . that there are no suspended threads" (Fig. 8; col. 17, lines 32-39; emphasis provided). First, the periodic thread check of Klemm is a *pre-test process* simply to determine whether "there was *previously* a *suspended thread* or an unexpired *probation* interval" *before the test is commenced* (Fig. 8; Col. 17, lines 32-39; emphasis provided). Klemm does not teach or reasonably suggest the input file having a sleep loop command to cause execution of the first program to pause the execution for a specified time, as recited by claim 1.

Second, the periodic thread check is *to check* for suspended threads and unexpired probation intervals. This is not the same as to pause the execution for a specified time, as recited by claim 1. Hence, Shrader and Klemm, neither individually nor when combined, teach or reasonably suggest the input file having a sleep loop command to cause execution of the first program to pause the execution for a specified time, as recited by claim 1. Accordingly, for at least the reasons set forth above, Appellants respectfully submit that claim 1 and its dependent claims are allowable over the cited references.

With regard to independent claims 13 and 25, these claims contain limitations similar to those of claim 1. Accordingly, Appellants respectfully submit that claims 13 and 25 and their dependent claims are allowable over the cited references.

With regard to claims 5, 7, 11-12, 17, 19, 23-24, 29, 31 and 35-36, these claims depend from one of independent claims 1, 13 and 25 and thus include the limitations of the independent claim from which they depend. Accordingly, Appellants respectfully submit that claims 5, 7, 11-12, 17, 19, 23-24, 29, 31 and 35-36 are allowable over the cited references.

IX. CONCLUSION

Appellants respectfully submit that all the appealed claims in this application are patentable and request that the Board of Patent Appeals and Interferences overrule the Examiner and direct allowance of the rejected claims.

This brief is submitted in triplicate, along with a check for $330.00 to cover the appeal fee for one other than a small entity as specified in 37 C.F.R. § 1.17(c). Please charge any shortages and credit any overpayment to out Deposit Account No. 02-2666.

Respectfully submitted,

BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN

Date: August 19, 2004

Aslam A. Jaffery
Attorney for Appellants
Reg. No. 51,841

12400 Wilshire Boulevard
Seventh Floor
Los Angeles, CA 90025-1030
(303) 740-1980

APPENDIX OF CLAIMS
(37 C.F.R. § 1.192(c)(9))

1. A method of testing a first program, comprising:

reading an input file containing a location of a first program, identifiers of other programs to be invoked by the first program, and arguments to be passed to the other programs, wherein the input file further includes special commands having

a loop command to cause execution of the first program including repeating of instructions for a number of iterations until occurrence of an endloop instruction, and

a sleep command to cause execution of the first program to pause for a specified time;

executing the first program including invoking the other programs and passing the arguments to the other programs; and

generating one or more log files based on results of the execution of the first program.

Claims 2-3 (Cancelled)

4. The method of claim 1, wherein said input file further includes a text file.

5. The method of claim 1, wherein said first program includes a Java Bean.

6. The method of claim 1, wherein said other programs include an Application Program Interface (API).

7. The method of claim 1, wherein the executing of the first program further comprises sending commands to an Automatic Call Distributor (ACD) switch through a CTI server.

8. The method of claim 1, wherein each of said one or more log files includes a text file.

9. The method of claim 1, wherein each of said one or more log files includes a plurality of entries, each entry of the plurality of entries having a date and time stamp.

10. The method of claim 1, wherein each of said one or more log files includes contents that are based on the results of the execution of the first program and the other programs.

11. The method of claim 10, wherein the contents are further based on the results of the execution of the first program and responses from an Automatic Call Distributor (ACD) switch.

12. The method of claim 1, further comprising checking results of program testing by:

generating one or more files containing expected results of the execution of the first program and the other programs; and

comparing the one or more files containing the expected results to said one or more log files.

13. A system, comprising:

a storage device having stored therein a routine for testing a first program; and

a processor coupled to the storage device, the processor to execute the routine for testing of the first program such that execution of said routine causes the processor to:

read an input file containing a location of the first program, identifiers of other programs to be invoked by the first program, and arguments to be passed to the other programs, wherein the input file further includes special commands having

a loop command to cause execution of the first program including repeating of instructions for a number of iterations until occurrence of an endloop instruction, and

a sleep command to cause execution of the first program to pause for a specified time,

execute the first program including invoking the other programs and passing the arguments to the other programs, and

generate one or more log files based on results of the execution of the first program.

Claims 14-15 (Cancelled)

16. The system of claim 13, wherein said input file includes a text file.

17. The system of claim 13, wherein said first program includes a Java Bean.

18. The system of claim 13, wherein said other programs include an Application Program Interface (API).

19. The system of claim 13, wherein the execution of the first program further comprises making calls to an Automatic Call Distribution (ACD) switch.

20. The system of claim 13, wherein each of said one or more log files includes a text file.

21. The system of claim 13, wherein each of said on or more log files includes a plurality of entries, each of the plurality of entries including a date and time stamp.

22. The system of claim 13, wherein each of said one or more log files includes contents that are based on the results of the execution of the first program and the other programs.

23. The system of claim 22, wherein the contents are further based on the results of the execution of the first program and responses from an Automatic Call Distributor (ACD) switch.

24. The system of claim 13, further comprising instructions for checking results of program testing such that execution of the instructions causes the processor to:

generate one or more files containing expected results of the execution of the first program and the other programs; and

compare the one or more files containing the expected results to the one or more log files.

25. A machine-readable medium having stored thereon sets of instructions which, when executed by a machine, test a first program by causing the machine to:

read an input file containing a location of the first program, identifiers of other programs to be invoked by the first program, and arguments to be passed to the other programs, wherein the input file further includes special commands having

a loop command to cause execution of the first program including repeating of instructions for a number of iterations until occurrence of an endloop instruction, and

a sleep command to cause execution of the first program to pause for a specified time;

execute the first program including invoking the other programs listed and passing the arguments to the other programs; and generate one or more log files based on results of execution of the first program.

Claims 26-27 (Cancelled)

28. The machine-readable medium of claim 25, wherein said input file further includes a text file.

29. The machine-readable medium of claim 25, wherein said first program includes a Java Bean.

30. The machine-readable medium of claim 25, wherein said other programs include an Application Program Interface (API).

31. The machine-readable medium of claim 25, wherein the executing of the first program further comprises making calls to an Automatic Call Distributor (ACD) switch.

32. The machine-readable medium of claim 25, wherein each of said one or more log files includes a text file.

33. The machine-readable medium of claim 25, wherein each of said one or more log files includes a plurality of entries, each entry of the plurality of entries having a date and time stamp.

34. The machine-readable medium of claim 25, wherein each of said one or more log files includes contents that are based on the results of the execution of the first program and the other programs.

35. The machine-readable medium of claim 34, wherein the contents are further based on the results of the execution of the first program and responses from an Automatic Call Distributor (ACD) switch.

36. The machine-readable medium of claim 25, further comprises instructions for checking results of program testing such that execution of the instructions causes the machine to:

generate one or more files containing expected results of the execution of the first program and the other programs; and

compare the one or more files containing the expected results said one or more log files.